METHOD AND APPARATUS FOR PROVIDING INTERPROCESSOR COMMUNICATIONS USING SHARED MEMORY

TECHNICAL FIELD

This invention relates in general to the field of electronics and more specifically to a method and apparatus for providing interprocessor communications (IPC) using shared memory.

BACKGROUND

Prior art techniques for sharing memory used for exchanging messages between two or more processors in an electronic system typically require that the two or more processors be responsible for their own "transmit memory" (memory used by a processor to load data that will be transmitted to another processor). Each processor is responsible for allocating and freeing message memory used for storing messages sent to the other processor(s). These prior art techniques force the static division of shared IPC memory between the two or more processors, meaning that a predetermined amount of the shared memory will need to be allocated to each processor. This may create a suboptimal use of the total shared memory that is available if the transmission of messages between the processors is asymmetrical (e.g., one processor sends more messages than another processor). With pre-allocated memory schemes, one processor's shared memory allocation may be under utilized while a second processor's shared memory allocation may not be enough for its message transmission needs. Given the above, a need exists in the art for a method

and apparatus which can help improve the sharing of memory between two or more processors.

BRIEF DESCRIPTION OF THE DRAWINGS

The features of the present invention, which are believed to be novel, are set forth with particularity in the appended claims. The invention may best be understood by reference to the following description, taken in conjunction with the accompanying drawings, in the several figures of which like reference numerals identify like elements, and in which:

FIG. 1 shows a diagram highlighting a method of multiprocessor sharing of memory in accordance with an embodiment of the invention.

FIG. 2 shows a block diagram of a radio communication device in accordance with an embodiment of the invention.

FIG. 3 shows a flow chart highlighting the steps taken by a first ("master") processor to transfer a message to a second processor in accordance with an embodiment of the invention.

Fig. 4 shows a flow chart highlighting the steps taken by a second ("non-master") processor to send a message to a first processor in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

While the specification concludes with claims defining the features of the invention that are regarded as novel, it is believed that the invention will be better

understood from a consideration of the following description in conjunction with the drawing figures.

In order to overcome the problems previously mentioned with some prior art IPC communications, the "transmit" memories of two or more processors are combined into one memory space managed by one of the processors in the system. In FIG. 1, there is shown a diagram illustrating the transfer of messages between a first ("master') processor (processor 1)102 and a second processor (processor 2) 104. The first processor 102 allocates memory among the processor 102 and 104, as the need for memory arises. The processors 102 and 104 can comprise any type of processor such as a microprocessor, microcontroller, or digital signal processor (DSP).

When the second processor 104 needs to send a message to the first processor 102, it sends an IPC empty message buffer request message as shown in step 106 to the first processor 102. The first processor 102 responds by sending an IPC empty message buffer pointer to the second processor 104 (step108). The empty message buffer pointer provides memory address information needed by the second processor 104 when accessing shared memory 112. The pointer informs the second processor 104 where in shared memory 112 it needs to start loading its message. Shared memory 112 can comprise Random Access Memory (RAM) or any other type of readable/writable memory known in the art.

The second processor 104 fills up the assigned message buffer 114 found in shared memory 112 and passes the message pointer back to the first processor 102 in step 110 so that it can read (consume) the data and free the previously assigned message buffer 114. Step 110 can include, in one embodiment, simply sending the message buffer pointer back to the first processor 102. In an alternate embodiment,

the second processor 104 can send another type of message to the first processor 102 which lets it know which message buffer (in this example IPC message buffer 114) was assigned to the second processor 104.

In order to reduce the latency of the second processor 104 asking for a message buffer from the first processor 102, in an alternate embodiment of the invention, a small set of buffers 116 is made available all the time to the second processor 104. Buffers 116 are ready to be used without the need for the second processor 104 requesting the buffers 116 from the first processor 102. Once a buffer from the assigned buffers 116 is removed for use by the second processor 104, the second processor 104 sends a message to the first processor 102 which automatically replaces the buffer when it receives the message.

Referring to FIG. 2, there is shown an electronic device such as a radio communication device 200 in accordance with the invention. A first processor (processor #1) 202 is coupled to a second processor (processor #2) 204 and both processors are coupled to shared memory 206. A conventional transmitter and receiver section 208 provides for radio frequency transmissions of messages. User controls (e.g., keypad) 210 and display 212 provide an interface to the user of the radio communication device 200.

In another embodiment of the invention, a mailbox buffer such as a one word mailbox 218 can be used to store the memory buffer pointer in the second processor 204 that is sent by the first processor 202. A similar mailbox, mailbox 216, can be found in the first processor 202.

The mailboxes 216 and 218 are used to exchange pointers and short commands between the first 202 and second processors 204. Alternatively, an

interrupt line 214 can be used by the first processor 202 to send an interrupt to the second processor 204. In response to receiving the interrupt, the second processor 204 reads a predetermined location in shared memory and locates the address pointer for the message buffer found in shared memory 206

In FIG. 3, there is shown a flowchart highlighting the steps taken by the first processor 202 when sending a message to the second processor 204. In this example, the first processor 202 is the master processor in charge of memory allocation for the shared memory 206. In step 302, the first processor (processor 1) 202 allocates memory from shared memory 206 for a message it needs to transfer (transmit) to the second processor 204. In step 304, the first processor 202 loads the message in the allocated memory area. In step 306, the first processor 202 sends the pointer to the second processor 204. In step 308, the second processor 204 receives the message from the shared memory 206. In step 310, after the message is consumed by the second processor 204, the second processor 204 sends a message pointer to the first processor 202 (using a mailbox 216 or interrupt line 214) indicating that the message space can be released. Finally in step 312, the first processor 202 releases the allocated memory to the shared memory 206. The release of the allocated memory can be performed by the second processor 204, for example, by sending a message to the first processor 202 that it has received the message.

In FIG. 4, there is shown a flowchart highlighting the steps taken by the second processor 204 when requesting memory space for the transmission of a message to the first processor 202. In step 402, the second processor 204 sends a request for memory allocation to the first processor 202. In step 404, the first processor 202 allocates the required memory and sends a pointer to the second

processor 204. In step 406, the second processor 204 loads the message in the allocated memory area and sends back the pointer to the first processor 202. Finally, in step 408, the first processor 202 retrieves the message from the allocated memory area and after retrieving the message, releases the allocated memory back to the memory pool.

The present invention allows for the implementation of a shared memory scheme that optimizes memory usage and minimizes overhead during message transfers between processors. By doing away with the static allocation of memory common in the prior art, the shared memory scheme of the present invention provides for an efficient memory allocation technique and system.

While the preferred embodiments of the invention have been illustrated and described, it will be clear that the invention is not so limited. Numerous modifications, changes, variations, substitutions and equivalents will occur to those skilled in the art without departing from the spirit and scope of the present invention as defined by the appended claims.

What is claimed is: